

Aperiodic String Transducers [∗]

Luc Dartois¹, Ismaël Jecker¹, Pierre-Alain Reynier²

¹ Université Libre de Bruxelles, Belgium

² Aix-Marseille Université, CNRS, LIF UMR 7279, France

Abstract. Regular string-to-string functions enjoy a nice triple characterization through deterministic two-way transducers (2DFT), streaming string transducers (SST) and MSO definable functions. This result has recently been lifted to FO definable functions, with equivalent representations by means of *aperiodic* 2DFT and *aperiodic* 1-bounded SST, extending a well-known result on regular languages. In this paper, we give three direct transformations: *i*) from 1-bounded SST to 2DFT, *ii*) from 2DFT to copyless SST, and *iii*) from k -bounded to 1-bounded SST. We give the complexity of each construction and also prove that they preserve the aperiodicity of transducers. As corollaries, we obtain that FO definable string-to-string functions are equivalent to SST whose transition monoid is finite and aperiodic, and to aperiodic copyless SST.

1 Introduction

The theory of regular languages constitutes a cornerstone in theoretical computer science. Initially studied on languages of finite words, it has since been extended in numerous directions, including finite and infinite trees. Another natural extension is moving from languages to transductions. We are interested in this work in string-to-string transductions, and more precisely in string-to-string functions. One of the strengths of the class of regular languages is their equivalent presentation by means of automata, logic, algebra and regular expressions. The class of so-called *regular string functions* enjoys a similar multiple presentation. It can indeed be alternatively defined using deterministic two-way finite state transducers (2DFT), using Monadic Second-Order graph transductions interpreted on strings (MSOT) [8], and using the model of streaming string transducers (SST) [1]. More precisely, regular string functions are equivalent to different classes of SST, namely copyless SST [1] and k -bounded SST, for every positive integer k [3]. Different papers [8,1,3,2] have proposed transformations between 2DFT, MSOT and SST, summarized on Figure 1.

The connection between automata and logic, which has been very fruitful for model-checking for instance, also needs to be investigated in the framework of transductions. As it has been done for regular languages, an important objective is then to provide similar logic-automata connections for subclasses of regular functions, providing decidability results for these subclasses. As an illustration,

[∗] This work is supported by the ARC project Transform (French speaking community of Belgium), the Belgian FNRS PDR project Flare, and the PHC project VAST (35961QJ) funded by Campus France and WBI.

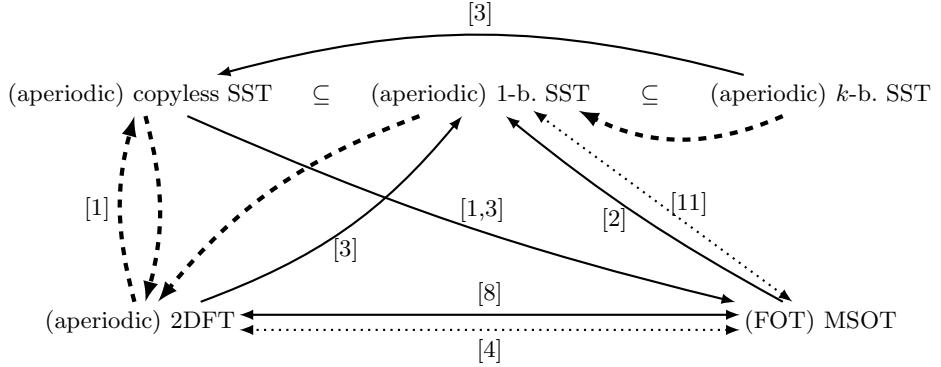


Fig. 1: Summary of transformations between equivalent models. k -b. stands for k -bounded. Plain (resp. dotted) arrows concern regular models (resp. bracketed models). Original constructions presented in this paper are depicted by thick dashed arrows and are valid for both regular and aperiodic versions of the models.

the class of rational functions (accepted by one-way finite state transducers) owns a simple characterization in terms of logic, as shown in [9]. The corresponding logical fragment is called order-preserving MSOT. The decidability of the one-way definability of a two-way transducer proved in [10] thus yields the decidability of this fragment inside the class of MSOT.

The first-order logic considered with order predicate constitutes an important fragment of the monadic second order logic. It is well known that languages definable using this logic are equivalent to those recognized by finite state automata whose transition monoid is aperiodic (as well as other models such as star-free regular expressions). These positive results have motivated the study of similar connections between first-order definable string transformations (FOT) and restrictions of state-based transducers models. Two recent works provide such characterizations for 1-bounded SST and 2DFT respectively [11,4]. To this end, the authors study a notion of transition monoid for these transducers, and prove that FOT is expressively equivalent to transducers whose transition monoid is aperiodic by providing back and forth transformations between FOT and 1-bounded aperiodic SST (resp. aperiodic 2DFT). In particular, [11] lets as an open problem whether FOT is also equivalent to aperiodic copyless SST and to aperiodic k -bounded SST, for every positive integer k . It is also worth noticing that these characterizations of FOT, unlike the case of languages, do not allow to decide the class FOT inside the class MSOT. Indeed, while decidability for languages relies on the syntactic congruence of the language, no such canonical object exists for the class of regular string transductions.

In this work, we aim at improving our understanding of the relationships between 2DFT and SST. We first provide an original transformation from 1-bounded (or copyless) SST to 2DFT, and study its complexity. While the existing construction used MSO transformations as an intermediate formalism, resulting in a non-elementary complexity, our construction is in double exponential time,

and in single exponential time if the input SST is copyless. Conversely, we describe a direct construction from 2DFT to copyless SST, which is similar to that of [1], but avoids the use of an intermediate model. These constructions also allow to establish links between the crossing degree of a 2DFT, and the number of variables of an equivalent copyless (resp. 1-bounded) SST, and conversely. Last, we provide a direct construction from k -bounded SST to 1-bounded SST, while the existing one was using copyless SST as a target model and not 1-bounded SST [3]. These constructions are represented by thick dashed arrows on Figure 1.

In order to lift these constructions to aperiodic transducers, we introduce a new transition monoid for SST, which is intuitively more precise than the existing one (more formally, the existing one divides the one we introduce). We use this new monoid to prove that the three constructions we have considered above preserve the aperiodicity of the transducer. As a corollary, this implies that FOT is equivalent to both aperiodic copyless and k -bounded SST, for every integer k , two results that were stated as conjectures in [11] (see Figure 1).

2 Definitions

2.1 Words, Languages and Transducers

Given a finite alphabet A , we denote by A^* the set of finite words over A , and by ϵ the empty word. The length of a word $u \in A^*$ is its number of symbols, denoted by $|u|$. For all $i \in \{1, \dots, |u|\}$, we denote by $u[i]$ the i -th letter of u .

A *language* over A is a set $L \subseteq A^*$. Given two alphabets A and B , a *transduction* from A to B is a relation $R \subseteq A^* \times B^*$. A transduction R is *functional* if it is a function. The transducers we will introduce will define transductions. We will say that two transducers T, T' are equivalent whenever they define the same transduction.

Automata A *deterministic two-way finite state automaton* (2DFA) over a finite alphabet A is a tuple $\mathcal{A} = (Q, q_0, F, \delta)$ where Q is a finite set of states, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is a set of final states, and δ is the transition function, of type $\delta : Q \times (A \uplus \{\vdash, \dashv\}) \rightarrow Q \times \{+1, 0, -1\}$. The new symbols \vdash and \dashv are called *endmarkers*.

An input word u is given enriched by the endmarkers, meaning that \mathcal{A} reads the input $\vdash u \dashv$. We set $u[0] = \vdash$ and $u[|u| + 1] = \dashv$. Initially the head of \mathcal{A} is on the first cell \vdash in state q_0 (the cell at position 0). When \mathcal{A} reads an input symbol, depending on the transitions in Δ , its head moves to the left (-1), or stays at the same position (0), or moves to the right ($+1$). To ensure the fact that the reading of \mathcal{A} does not go out of bounds, we assume that there is no transition moving to the left (resp. to the right) on input symbol \vdash (resp. \dashv). \mathcal{A} stops as soon as it reaches the endmarker \dashv in a final state.

A *configuration* of \mathcal{A} is a pair $(q, i) \in Q \times \mathbb{N}$ where q is a state and i is a position on the input tape. A *run* ρ of \mathcal{A} is a finite sequence of configurations. The run $\rho = (p_1, i_1) \dots (p_m, i_m)$ is a run on an input word $u \in A^*$ of length n if $i_m \leq n+1$, and for all $k \in \{1, \dots, m-1\}$, $0 \leq i_k \leq n+1$ and $(p_k, u[i_k], p_{k+1}, i_{k+1} - i_k) \in \Delta$.

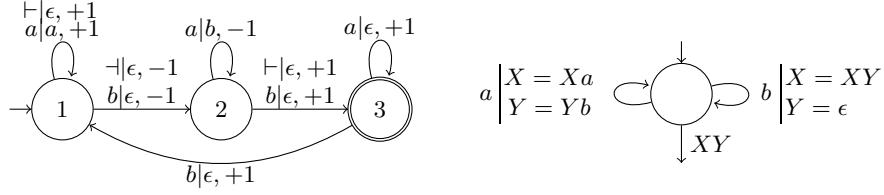


Fig. 2: Aperiodic 2DFT (left) and SST (right) realizing the function f .

It is *accepting* if $p_1 = q_0$, $i_1 = 0$, and m is the only index where both $i_m = n + 1$ and $p_m \in F$. The language of a 2DFA \mathcal{A} , denoted by $L(\mathcal{A})$, is the set of words u such that there exists an accepting run of \mathcal{A} on u .

Transducers *Deterministic two-way finite state transducers* (2DFT) over A extend 2DFA with a one-way left-to-right output tape. They are defined as 2DFA except that the transition relation δ is extended with outputs: $\delta : Q \times (A \uplus \{\vdash, \dashv\}) \rightarrow B^* \times Q \times \{-1, 0, +1\}$. When a transition (q, a, v, q', m) is fired, the word v is appended to the right of the output tape.

A run of a 2DFT is a run of its underlying automaton, i.e. the 2DFA obtained by ignoring the output (called its *underlying input automaton*). A run ρ may be simultaneously a run on a word u and on a word $u' \neq u$. However, when the input word is given, there is a unique sequence of transitions associated with ρ . Given a 2DFT T , an input word $u \in A^*$ and a run $\rho = (p_1, i_1) \dots (p_m, i_m)$ of T on u , the output of ρ on u is the word obtained by concatenating the outputs of the transitions followed by ρ . If ρ contains a single configuration, this output is simply ϵ . The transduction defined by T is the relation $R(T)$ defined as the set of pairs $(u, v) \in A^* \times B^*$ such that v is the output of an accepting run ρ on the word u . As T is deterministic, such a run is unique, thus $R(T)$ is a function.

Streaming String Transducers Let \mathcal{X} be a finite set of variables denoted by X, Y, \dots and B be a finite alphabet. A substitution σ is defined as a mapping $\sigma : \mathcal{X} \rightarrow (B \cup \mathcal{X})^*$. Let $\mathcal{S}_{\mathcal{X}, B}$ be the set of all substitutions. Any substitution σ can be extended to $\hat{\sigma} : (B \cup \mathcal{X})^* \rightarrow (B \cup \mathcal{X})^*$ in a straightforward manner. The composition $\sigma_1 \sigma_2$ of two substitutions σ_1 and σ_2 is defined as the standard function composition $\hat{\sigma}_1 \hat{\sigma}_2$, i.e. $\hat{\sigma}_1 \hat{\sigma}_2(X) = \hat{\sigma}_1(\hat{\sigma}_2(X))$ for all $X \in \mathcal{X}$. We say that a string $u \in (B \cup \mathcal{X})^*$ is *k-linear* if each $X \in \mathcal{X}$ occurs at most k times in u . A substitution σ is *k-linear* if $\sigma(X)$ is *k-linear* for all X . It is *copyless* if for any variable X , there exists at most one variable Y such that X occurs in $\sigma(Y)$, and X occurs at most once in $\sigma(Y)$.

A *streaming string transducer* (SST) is a tuple $T = (A, B, Q, q_0, Q_f, \delta, \mathcal{X}, \rho, F)$ where (Q, q_0, Q_f, δ) is a one-way automaton, A and B are finite sets of input and output alphabets respectively, \mathcal{X} is a finite set of variables, $\rho : \delta \rightarrow \mathcal{S}_{\mathcal{X}, B}$ is a variable update and $F : Q_f \rightarrow (\mathcal{X} \cup B)^*$ is the output function.

Example 1. As an example, let $f : \{a, b\}^* \rightarrow \{a, b\}^*$ be the function mapping any word $u = a^{k_0}ba^{k_1} \dots ba^{k_n}$ to the word $f(u) = a^{k_0}b^{k_0}a^{k_1}b^{k_1} \dots a^{k_n}b^{k_n}$ obtained by adding after each block of consecutive a a block of consecutive b of the same length. Since each word u over A can be uniquely written $u = a^{k_0}ba^{k_1} \dots ba^{k_n}$

with some k_i being possibly equal to 0, the function f is well defined. We give in Figure 2 a 2DFT and an SST that realize f .

The concept of a run of an SST is defined in an analogous manner to that of a finite state automaton. The sequence $\langle \sigma_{r,i} \rangle_{0 \leq i \leq |r|}$ of substitutions induced by a run $r = q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \dots q_{n-1} \xrightarrow{a_n} q_n$ is defined inductively as the following: $\sigma_{r,i} = \sigma_{r,i-1} \rho(q_{i-1}, a_i)$ for $1 < i \leq |r|$ and $\sigma_{r,1} = \rho(q_0, a_1)$. We denote $\sigma_{r,|r|}$ by σ_r and say that σ_r is induced by r .

If r is accepting, i.e. $q_n \in Q_f$, we can extend the output function F to r by $F(r) = \sigma_\epsilon \sigma_r F(q_n)$, where σ_ϵ substitutes all variables by their initial value ϵ . For all words $u \in A^*$, the output of u by T is defined only if there exists an accepting run r of T on u , and in that case the output is denoted by $T(u) = F(r)$. The transformation $R(T)$ is then defined as the set of pairs $(u, T(u)) \in A^* \times B^*$.

An SST T is copyless if for every transition $t \in \delta$, the variable update $\rho(t)$ is copyless. Given an integer $k \in \mathbb{N}_{>0}$, we say that T is k -bounded if all its runs induce k -linear substitutions. It is *bounded* if it is k -bounded for some k .

The following theorem gives the expressiveness equivalence of the models we consider. We do not give the definitions of MSO graph transductions as our results will only involve state-based transducers (see [9] for more details).

Theorem 1 ([8,1,3]). *Let $f : A^* \rightarrow B^*$ be a function over words. Then the following conditions are equivalent:*

- f is realized by an MSO graph transduction,
- f is realized by a 2DFT,
- f is realized by a copyless SST,
- f is realized by a bounded SST.

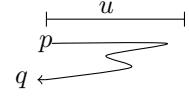
2.2 Transition monoid of transducers

A (finite) monoid M is a (finite) set equipped with an associative internal law \cdot_M having a neutral element for this law. A morphism $\eta : M \rightarrow N$ between monoids is an application from M to N that preserves the internal laws, meaning that for all x and y in M , $\eta(x \cdot_M y) = \eta(x) \cdot_N \eta(y)$. When the context is clear, we will write xy instead of $x \cdot_M y$. A monoid M divides a monoid N if there exists an onto morphism from a submonoid of N to M . A monoid M is said to be *aperiodic* if there exists a least integer n , called the *aperiodicity index* of M , such that for all elements x of M , we have $x^n = x^{n+1}$.

Given an alphabet A , the set of words A^* is a monoid equipped with the concatenation law, having the empty word as neutral element. It is called the *free monoid* on A . A finite monoid M *recognizes* a language L of A^* if there exists an onto morphism $\eta : A^* \rightarrow M$ such that $L = \eta^{-1}(\eta(L))$. It is well-known that the languages recognized by finite monoids are exactly the regular languages.

The monoid we construct from a machine is called its *transition monoid*. We are interested here in aperiodic machines, in the sense that a machine is aperiodic if its transition monoid is aperiodic. We now give the definition of the transition monoid for a 2DFT and an SST.

Deterministic Two-Way Finite State Transducers As in the case of automata, the transition monoid of a 2DFT T is the set of all possible behaviors of T on a word. The following definition comes from [4], using ideas from [14] amongst others. As a word can be read in both ways, the possible runs are split into four relations over the set of states Q of T . Given an input word u , we define the left-to-left behavior $\text{bh}_{\ell\ell}(u)$ as the set of pairs (p, q) of states of T such that there exists a run over u starting on the first letter of u in state p and exiting u on the left in state q (see Figure on the right). We define in an analogous fashion the left-to-right, right-to-left and right-to-right behaviors denoted respectively $\text{bh}_{\ell r}(u)$, $\text{bh}_{r\ell}(u)$ and $\text{bh}_{rr}(u)$. Then the transition monoid of a 2DFT is defined as follows:



Let $T = (Q, A, \delta, q_0, F)$ be a 2DFT. The *transition monoid* of T is A^*/\sim_T where \sim_T is the conjunction of the four relations \sim_{ll} , \sim_{lr} , \sim_{rl} and \sim_{rr} defined for any words u, u' of A^* as follows: $u \sim_{xy} u'$ iff $\text{bh}_{xy}(u) = \text{bh}_{xy}(u')$, for $x, y \in \{\ell, r\}$. The neutral element of this monoid is the class of the empty word ϵ , whose behaviors $\text{bh}_{xy}(\epsilon)$ is the identity function if $x \neq y$, and is the empty relation otherwise.

Note that since the set of states of T is finite, each behavior relation is of finite index and consequently the transition monoid of T is also finite. Let us also remark that the transition monoid of T does not depend on the output and is in fact the transition monoid of the underlying 2DFA.

Streaming String Transducers A notion of transition monoid for SST was defined in [11]. We give here its formal definition and refer to [11] for advanced considerations. In order to describe the behaviors of an SST, this monoid describes the possible flows of variables along a run. Since we give later an alternative definition of transition monoid for SST, we will call it the *flow transition monoid* (FTM).

Let T be an SST with states Q and variables \mathcal{X} . The *flow transition monoid* M_T of T is a set of square matrices over the integers enriched with a new absorbent element \perp . The matrices are indexed by elements of $Q \times \mathcal{X}$. Given an input word u , the image of u in M_T is the matrix m such that for all states p, q and all variables X, Y , $m[p, X][q, Y] = n \in \mathbb{N}$ (resp. $m[p, X][q, Y] = \perp$) if, and only if, there exists a run r of T over u from state p to state q , and X occurs n times in $\sigma_r(Y)$ (resp. iff there is no run of T over u from state p to state q).

Note that if T is k -bounded, then for all word w , all the coefficients of its image in M_T are bounded by k . The converse also holds. Then M_T is finite if, and only if, T is k -bounded, for some k .

It can be checked that the machines given in Example 1 are aperiodic. Theorem 1 extends to aperiodic subclasses and to first-order logic, as in the case of regular languages [13,12]. These results as well as our contributions to these models are summed up in Figure 1.

Theorem 2 ([11,4]). *Let $f : A^* \rightarrow B^*$ be a function over words. Then the following conditions are equivalent:*

- f is realized by a FO graph transduction,

- f is realized by an aperiodic 2DFT,
- f is realized by an aperiodic 1-bounded SST.

3 Substitution Transition Monoid

In this section, we give an alternative take on the definition of the transition monoid of an SST, and show that both notions coincide on aperiodicity and boundedness. The intuition for this monoid, that we call the *substitution transition monoid*, is for the elements to take into account not only the multiplicity of the output of each variable in a given run, but also the order in which they appear in the output. It can be seen as an enrichment of the classic view of transition monoids as the set of functions over states equipped with the law of composition. Given a substitution $\sigma \in \mathcal{S}_{\mathcal{X}, B}$, let us denote $\tilde{\sigma}$ the projection of σ on the set \mathcal{X} , i.e. we forget the parts from B . The substitutions $\tilde{\sigma}$ are homomorphisms of \mathcal{X}^* which form an (infinite) monoid. Note that in the case of a 1-bounded SST, each variable occurs at most once in $\tilde{\sigma}(Y)$.

Substitution Transition Monoid of an SST. Let T be an SST with states Q and variables \mathcal{X} . The *substitution transition monoid* (STM) of T , denoted M_T^σ , is a set of partial functions $f : Q \rightharpoonup Q \times \mathcal{S}_{\mathcal{X}, \emptyset}$. Given an input word u , the image of u in M_T^σ is the function f_u such that for all states p , $f_u(p) = (q, \tilde{\sigma}_r)$ if, and only if, there exists a run r of T over u from state p to state q that induces the substitution $\tilde{\sigma}_r$. This set forms a monoid when equipped with the following composition law: Given two functions $f_u, f_v \in M_T^\sigma$, the function f_{uv} is defined by $f_{uv}(q) = (q'', \tilde{\sigma} \circ \tilde{\sigma}')$ whenever $f_u(q) = (q', \tilde{\sigma})$ and $f_v(q') = (q'', \tilde{\sigma}')$.

We now make a few remarks about this monoid. Let us first observe that the FTM of T can be recovered from its STM. Indeed, the matrix m associated with a word u in M_T is easily deduced from the function f_u in M_T^σ . This observation induces an onto morphism from M_T^σ to M_T , and consequently the FTM of an SST divides its STM. This proves that if the STM is aperiodic, then so is the FTM since aperiodicity is preserved by division of monoids. Similarly, copyless and k -bounded SST (given $k \in \mathbb{N}_{>0}$) are characterized by means of their STM. This transition monoid can be separated into two main components: the first one being the transition monoid of the underlying deterministic one-way automaton, which can be seen as a set of functions $Q \rightarrow Q$, while the second one is the monoid $\mathcal{S}_{\mathcal{X}}$ of homomorphisms on \mathcal{X} , equipped with the composition. The aware reader could notice that the STM can be written as the wreath product of the transformation semigroup $(\mathcal{X}^*, \mathcal{S}_{\mathcal{X}})$ by (Q, Q^Q) . However, as the monoid of substitution is obtained through the closure under composition of the homomorphisms of a given SST, it may be infinite.

The next theorem proves that aperiodicity for both notions coincide, since the converse comes from the division of STM by FTM.

Theorem 3. *Let T be a k -bounded SST with ℓ variables. If its FTM is aperiodic with aperiodicity index n then its STM is aperiodic with aperiodicity index at most $n + (k + 1)\ell$.*

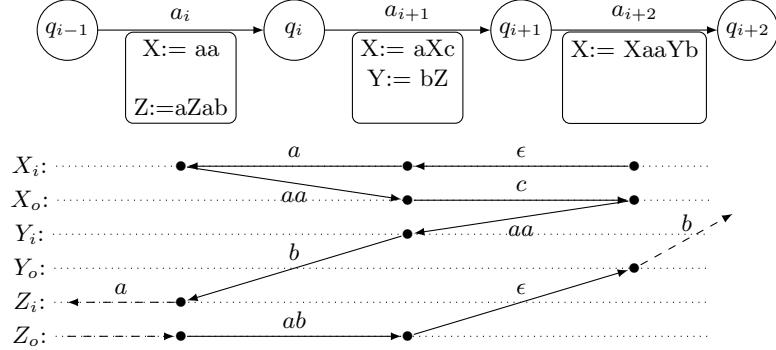


Fig. 3: The output structure of a partial run of an SST used in the proof of Theorem 4.

4 From 1-bounded SST to 2DFT

The existing transformation of a 1-bounded (or even copyless) SST into an equivalent 2DFT goes through MSO transductions, yielding a non-elementary complexity. We present here an original construction whose complexity is elementary.

Theorem 4. *Let T be a 1-bounded SST with n states and m variables. Then we can effectively construct a deterministic 2-way transducer that realizes the same function. If T is 1-bounded (resp. copyless), then the 2DFT has $O(m2^{m2^m}n^n)$ states (resp. $O(mn^n)$).*

Proof. We define the 2DFT as the composition of a left-to-right sequential transducer, a right-to-left sequential transducer and a 2-way transducer. Remark that this proves the result as two-way transducers are closed under composition with sequential ones [5].

The left-to-right sequential transducer does a single pass on the input word and outputs the same word enriched with the transition used by the SST in the previous step. The right-to-left transducer uses this information to enrich each position of the input word with the set of useful variables, i.e the variables that flow to an output variable according to the partial run on the suffix read. The two sequential transducers are quite standard. They realize length-preserving functions that simply enrich the input word with new information. The last transducer is more interesting: it uses the enriched information to follow the output structure of T . The *output structure* of a run is a labeled and directed graph such that, for each variable X useful at a position j , we have two nodes X_i^j and X_o^j linked by a path whose concatenated labels form the value stored in X at position j of the run (see [11] and Figure 3).

The transition function of the two-way transducer is described in Figure 4. It first reaches the end of the word and picks the first variable to output. It then rewinds the run using the information stored by the first sequential transducer, producing the said variable using the local update function. When it has finished

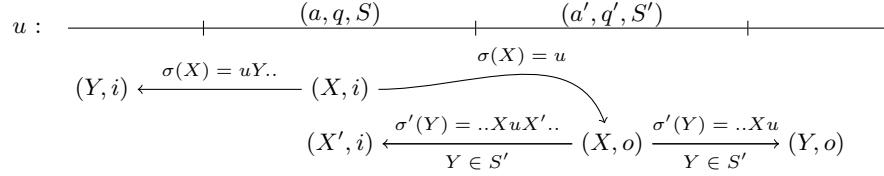


Fig. 4: The third transducer follows the output structure. States indexed by i correspond to the beginning of a variable, while states indexed by o correspond to the end. σ (resp. σ') stand for the substitution at position a (resp. a').

to compute and produce a variable X , it switches to the following one using the information of the second transducer to know which variable Y X is flowing to, and starts producing it. Note that such a Y is unique thanks to the 1-boundedness property. If T is copyless, then this information is local and the second transducer can be bypassed.

Regarding complexity, a careful analysis of the composition of a one-way transducer of size n with a two-way transducer of size m from [6,4] shows that this can be done by a two-way transducer of size $O(mn^n)$. Then given a 1-bounded SST with n states and m variables, we can construct a deterministic two-way transducer of size $O(m2^{m^2}n^n)$. If T is copyless, the second sequential transducer is omitted, resulting in a size of $O(mn^n)$.

Theorem 5. *Let T be an aperiodic 1-bounded SST. Then the equivalent 2DFT constructed using Theorem 4 is also aperiodic.*

Proof. The aperiodicity of the three transducers gives the result as aperiodicity is preserved by composition of a one-way by a two-way [4]. The aperiodicity of the two sequential transducers is straightforward since their runs depend respectively on the underlying automaton and the update function. The aperiodicity of the 2DFT comes from the fact that since it follows the output structure of the SST, its partial runs are induced by the flow of variables and their order in the substitutions, which is an information contained in the FTM and thus aperiodic thanks to Theorem 3.

5 From 2DFT to copyless SST

In [1], the authors give a procedure to construct a copyless SST from a 2DFT. This procedure uses the intermediate model of heap based transducers. We give here a direct construction with similar complexity. This simplified presentation allows us to prove that the construction preserves the aperiodicity.

Theorem 6. *Let T be a 2DFT with n states. Then we can effectively construct a copyless SST with $O((2n)^{2n})$ states and $2n - 1$ variables that computes the same function.*

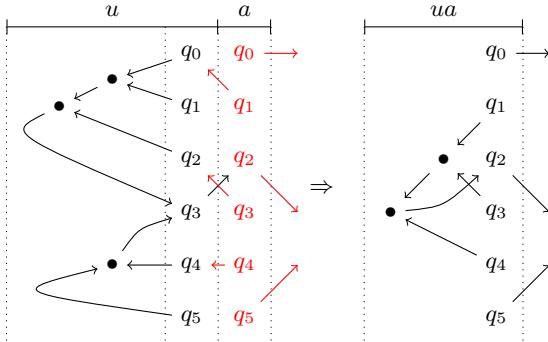


Fig. 5: Left: The state of the SST is represented in black. The red part corresponds to the local transitions of the 2DFT. Right: After reading a , we reduce the new forest by eliminating the useless branches and shortening the unlabeled linear paths.

Proof. (Sketch of) The main idea is for the constructed SST to keep track of the right-to-right behavior of the prefix read until the current position, similarly to the construction of Shepherdson [14]. This information can be updated upon reading a new letter, constructing a one-way machine recognizing the same input language. The idea from [3] is to have one variable per possible right-to-right run, which is bounded by the number of states. However, since two right-to-right runs from different starting states can merge, this construction results in a 1-bounded SST. To obtain copylessness, we keep track of these merges and the order in which they appear. Different variables are used to store the production of each run before the merge, and one more variable stores the production after.

The states of the copyless SST are represented by sets of labeled trees having the states of the input 2DFT as leaves. Each inner vertex represents one merging, and two leaves have a common ancestor if the right-to-right runs from the corresponding states merge at some point. Each tree then models a set of right-to-right runs that all end in a same state. Note that it is necessary to also store the end state of these runs. For each vertex, we use one variable to store the production of the partial run corresponding to the outgoing edge.

Given such a state and an input letter, the transition function can be defined by adding to the set of trees the local transitions at the given letter, and then reducing the resulting graph in a proper way (see Figure 5).

Finally, as merges occur upon two disjoint sets of states of the 2DFT (initially singletons), the number of merges, and consequently the number of inner vertices of our states, is bounded by $n - 1$. Therefore, an input 2DFT with n states can be realized by an SST having $2n - 1$ variables. Finally, as states are labeled graphs, Cayley's formula yields an exponential bound on the number of states.

Moreover, this construction preserves aperiodicity:

Theorem 7. *Let T be an aperiodic 2DFT. Then the equivalent SST constructed using Theorem 6 is also aperiodic.*

Proof. If the input 2DFT is aperiodic of index n , then for any word w , w^n and w^{n+1} merge the same partial runs for the four kinds of behaviors, by definition, and in fact the merges appear in the same order. As explained earlier, the state q_1 (resp. q_2) reached by the constructed SST over the inputs uw^n (resp. uw^{n+1}) represents the merges of the right-to-right runs of T over uw^n (resp. uw^{n+1}).

Since these runs can be decomposed in right-to right runs over u and partial runs over w^n and w^{n+1} , the merge equivalence between w^n and w^{n+1} implies that $q_1 = q_2$. Moreover, since variables are linked to these merges, the aperiodicity of the merge equivalence implies the aperiodicity of both the underlying automaton and the substitution function of the SST, concluding the proof.

As a corollary, we obtain that the class of aperiodic copyless SST is expressively equivalent to first-order definable string-to-string transductions.

Corollary 1. *Let $f : A^* \rightarrow B^*$ be a function over words. Then f is realized by a FO graph transduction iff it is realized by an aperiodic copyless SST.*

6 From k -bounded to 1-bounded SST

The existing construction from k -bounded to 1-bounded, presented in [3], builds a copyless SST. We present an alternative construction that, given a k -bounded SST, directly builds an equivalent 1-bounded SST. We will prove that this construction preserves aperiodicity.

Theorem 8. *Given a k -bounded SST T with n states and m variables, we can effectively construct an equivalent 1-bounded SST. This new SST has $n2^N$ states and mkN variables, where $N = O(n^n(k+1)^{nm^2})$ is the size of the flow transition monoid M_T .*

Proof. In order to move from a k -bounded SST to a 1-bounded SST, the natural idea is to use copies of each variable. However, we cannot maintain k copies of each variable all the time: suppose that X flows into Y and Z , which both occur in the final output. If we have k copies of X , we cannot produce in a 1-bounded way (and we do not need to) k copies of Y and k copies of Z .

Now, if we have access to a look-ahead information, we can guess how many copies of each variable are needed, and we can easily construct a copyless SST by using exactly the right number of copies for each variable and at each step. The construction relies on this observation. We simulate a look-ahead through a subset construction, having copies of each variable for each possible behavior of the suffix. Then given a variable and the behavior of a suffix, we can maintain the exact number of variables needed and perform a copyless substitution to a potential suffix for the next step. However, since the SST is not necessarily co-deterministic, a given suffix can have multiple successors, and the result is that its variables flow to variables of different suffixes. As variables of different suffixes are never recombined, we obtain a 1-bounded SST.

Theorem 9. *Let T be an aperiodic k -bounded SST. Then the equivalent 1-bounded SST constructed using Theorem 8 is also aperiodic.*

As a corollary, we obtain that for the class of aperiodic bounded SST is expressively equivalent to first-order definable string-to-string transductions.

Corollary 2. *Let $f : A^* \rightarrow B^*$ be a function over words. Then f is realized by a FO graph transduction iff it is realized by an aperiodic bounded SST ($k \in \mathbb{N}_{>0}$).*

7 Perspectives

There is still one model equivalent to the generic machines whose aperiodic sub-class elude our scope yet, namely the *functional two-way transducers*, which correspond to non-deterministic two-way transducers realizing a function. To complete the picture, a natural approach would then be to consider the constructions from [7] and prove that aperiodicity is preserved. One could also think of applying this approach to other varieties of monoids, such as the \mathcal{J} -trivial monoids, equivalent to the boolean closure of existential first-order formulas $\mathcal{B}\Sigma_1[<]$. Unfortunately, the closure of such transducers under composition requires some strong properties on varieties (at least closure under semidirect product) which are not satisfied by varieties less expressive than the aperiodic. Consequently the construction from SST to 2DFT cannot be applied. On the other hand, the other construction could apply, providing one inclusion. Then an interesting question would be to know where the corresponding fragment of logic would position.

References

1. R. Alur and P. Černý. Expressiveness of streaming string transducers. In *FSTTCS*, volume 8 of *LIPICS*, pages 1–12. Schloss Dagstuhl. Leibniz-Zent. Inform., 2010.
2. R. Alur, A. Durand-Gasselin, and A. Trivedi. From monadic second-order definable string transformations to transducers. In *LICS*, pages 458–467, 2013.
3. R. Alur, E. Filiot, and A. Trivedi. Regular transformations of infinite strings. In *LICS*, pages 65–74, 2012.
4. O. Carton and L. Dartois. Aperiodic two-way transducers and fo-transductions. In *CSL*, volume 41 of *LIPICS*, pages 160–174. Schloss Dagstuhl. Leibniz-Zent. Inform., 2015.
5. M. P. Chytil and V. Jákli. Serial composition of 2-way finite-state transducers and simple programs on strings. In *Automata, languages and programming*, pages 135–137. LNCS, Vol. 52. Springer, Berlin, 1977.
6. L. Dartois. *Méthodes algébriques pour la théorie des automates*. PhD thesis, LIAFA-Université Paris Diderot, Paris, 2014.
7. R. de Souza. Uniformisation of two-way transducers. In *LATA*, pages 547–558, 2013.
8. J. Engelfriet and H. J. Hoogeboom. MSO definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Log.*, 2(2):216–254, 2001.
9. E. Filiot. Logic-automata connections for transformations. In *ICLA*, volume 8923 of *LNCS*, pages 30–57. Springer, 2015.
10. E. Filiot, O. Gauwin, P.-A. Reynier, and F. Servais. From two-way to one-way finite state transducers. In *LICS*, pages 468–477. IEEE Computer Society, 2013.
11. E. Filiot, S. N. Krishna, and A. Trivedi. First-order definable string transformations. In *FSTTCS*, volume 29 of *LIPICS*, pages 147–159. Schloss Dagstuhl - Leibniz-Zent. Inform., 2014.
12. R. McNaughton and S. Papert. *Counter-free automata*. The M.I.T. Press, Cambridge, Mass.-London, 1971.
13. M. P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8:190–194, 1965.
14. J. C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3(2):198–200, 1959.

Appendix

Substitution Transition Monoid

Theorem 3. *Let T be a k -bounded SST with ℓ variables. If its FTM is aperiodic with aperiodicity index n then its STM is aperiodic with aperiodicity index at most $n + (k + 1)\ell$.*

Proof. Let T be a k -bounded SST. We define a *loop* as the run induced by a pair $(q, u) \in Q \times A^*$ such that $\delta(q, u) = q$. Suppose now that M_T is aperiodic, and let n be its aperiodicity index. Wlog, we assume that the transition function of T is complete. This implies that for all states p of T , there exists a state q such that $p \xrightarrow{u^n} q \xrightarrow{u} q$. Then if the image in STM of the loops (i.e. the set of all $\tilde{\sigma}$ such that there exists a loop (q, u) such that $f_u(q) = (q, \tilde{\sigma})$) are aperiodic with index m , then the STM is aperiodic with index at most $n + m$.

Consequently, in the following σ denotes the substitution of a loop of T , and we aim to prove that $\tilde{\sigma}^{(k+1)\ell} = \tilde{\sigma}^{(k+1)\ell+1}$.

Before proving this though, we define the relation $\lessdot \subseteq \mathcal{X} \times \mathcal{X}$ as follows. Given two variables X and Y , we have $X \lessdot Y$ if there exists a positive integer i such that X flows into Y in σ^i . This relation is clearly transitive. The next lemma proves that it is also anti-symmetric, hence we can use this relation as an induction order to prove the result.

Lemma 1. *Given two different variables X and Y , if $X \lessdot Y$, then $Y \not\lessdot X$.*

Proof. We proceed by contradiction. Assume that there exist two different variables X and Y and two integers i and j such that X occurs in $\sigma^i(Y)$ and Y occurs in $\sigma^j(X)$.

Then for any $k > 0$, X occurs in $\sigma^{k(i+j)}(X)$ and Y occurs in $\sigma^{k(i+j)+j}(X)$. As T is aperiodic of index n , for k large enough it means that both X and Y occur in both $\sigma^n(X)$ and $\sigma^n(Y)$. Then $\sigma^{2n}(X)$ contains both $\sigma^n(X)$ and $\sigma^n(Y)$ and thus contains at least two occurrences of X and Y . Then by aperiodicity we have $\sigma^{2n}(X) = \sigma^n(X)$ thus $\sigma^n(X)$ contains two occurrences of X . By iterating this process, we prove that the number of occurrences of X in $\sigma^n(X)$ is not bounded, yielding a contradiction.

We now prove that for all variables X in \mathcal{X} , $\tilde{\sigma}^{(k+1)\ell}(X) = \tilde{\sigma}^{(k+1)\ell+1}(X)$ by treating the following two cases:

- If $X \in \sigma(X)$, then either $\tilde{\sigma}(X) = X$ and then $\tilde{\sigma}^2(X) = \tilde{\sigma}(X)$, or there exists $Y \neq X$ such that $Y \in \tilde{\sigma}(X)$. In the latter case, we get by iteration that for all $i > 0$, $|\tilde{\sigma}^i(X)| > \sum_{j < i} |\tilde{\sigma}^j(Y)|$. Then as T is k -bounded, we have $|\tilde{\sigma}^i(X)| \leq k\ell$ and thus $\sum_{j < i} |\tilde{\sigma}^j(Y)|$ is bounded, and $\tilde{\sigma}^{k\ell}(Y) = \epsilon$, which proves that $\tilde{\sigma}^{k\ell}(X) = \tilde{\sigma}^{k\ell+1}(X)$.
- If $X \notin \sigma(X)$, let us consider the relation \lessdot . By Lemma 1 this relation is cycle-free. Then there is a lesser level, on which there are the variables Y such that $\tilde{\sigma}(Y) \subseteq \{Y\}$. There, either $\tilde{\sigma}(Y) = \emptyset$ and aperiodicity becomes trivial, or $\tilde{\sigma}(Y) = Y$ and the case was dealt with in the previous point and

is thus aperiodic with index $k\ell$. Now we can end the proof by reasoning by induction on \lessdot , as if $X \notin \sigma(X)$ and all variables $Y \lessdot X$ are aperiodic with index i , then $\tilde{\sigma}^{i+1}(X)$ can be written as the concatenation of $\tilde{\sigma}^i(Y)$, for aperiodic variables Y of index i . Then $\tilde{\sigma}^{i+1}(X) = \tilde{\sigma}^{i+2}(X)$. The proof is concluded by noticing that the length of the longest chain of \lessdot is bounded by ℓ .

From 1-bounded SST to 2DFT

Theorem 4. *Let T be a 1-bounded SST with n states and m variables. Then we can effectively construct a deterministic 2-way transducer that realizes the same function. If T is 1-bounded (resp. copyless), then the 2DFT has $O(m2^{m2^m}n^n)$ states (resp. $O(mn^n)$).*

Proof. Let $T = (A, B, Q, q_0, Q_f, \delta, \mathcal{X}, \rho, F)$ be a SST. Let us construct a two-way transducer \mathcal{A} that realizes the same function. The transducer \mathcal{A} will follow the output structure (see Figure 3) of T and construct the output as it appears in the structure. To make the proof easier to read, we define \mathcal{A} as the composition of a left-to-right sequential transducer \mathcal{A}' , a right-to-left sequential transducer \mathcal{A}'' and a 2-way transducer \mathcal{B} . Remark that this proves the result as two-way transducers are closed under composition with sequential ones. The transducer \mathcal{A}' does a single pass on the input and enriches it with the transition used by T in the previous step. The second transducer uses this information, and enriches the input word with the set of variables corresponding to the variables that will be produced from this position. The last transducer is more interesting: it uses the enriched information to follow the output structure of T . The *output structure* of a run is a labeled and directed graph such that, for each variable X useful at a position j , we have two nodes X_i^j and X_o^j linked by a path whose concatenated labels form the value stored in X at position j of the run (see [11] and Figure 3). The set of variables will be used to clear the non determinism due to the 1-bounded property. Note that in the case of a copyless SST, the transducer \mathcal{A}'' can be omitted. We now explain how the several transducers behave on a given run $r = q_0 \xrightarrow{a_1} q_1 \dots \xrightarrow{a_n} q_n$.

The transducer $\mathcal{A}' = (A, A \times Q, Q \times \{f\}, q_0, \alpha, \beta, \{f\})$, which enriches the input word with the transitions of the previous step, can be done easily with a 1-way transducer, which first stores the transition taken in its state, then outputs it along the current letter read. Given a letter a and a state q , we have the transitions $(q, a) \xrightarrow{a|(a,q)} \delta(q, a)$. We also have $(q, \dashv) \xrightarrow{\dashv|(\dashv,q)} f$. Then on the run r , if $u = a_1 \dots a_n$, we get the output word $\mathcal{A}'(\dashv u \dashv) = (a_1, q_0)(a_2, q_1) \dots (a_n, q_{n-1})(\dashv, q_n)$.

The transducer $\mathcal{A}'' = (A \times Q, A \times Q \times 2^{\mathcal{X}}, 2^{\mathcal{X}} \uplus \{i, f'\}, i, \alpha, \beta, \{f'\})$, which enriches each letter of the input word with the variables effectively produced from this step, can be done easily with a right-to-left sequential transducer, which starts with the variables appearing in $F(q_n)$. Given a letter (a, q) and a set S , we have $S \xrightarrow{(a,q)|(a,q,S)} S'$ where $S' = \{X \in \mathcal{X} \mid \exists Y \in S \text{ } X \text{ occurs in } \rho(q, a, Y)\}$.

Then given an input word $(a_1, q_0)(a_2, q_1) \dots (\dashv, q_n)$, we define $S_n = F(q_n)$, and $S_{i-1} = \{X \in \mathcal{X} \mid \exists Y \in F(q_n) \ X \in \sigma_{q_{i-1}, a_i \dots a_n}(Y)\}$, we get the output word $\mathcal{A}'' \circ \mathcal{A}'(\dashv u \dashv) = (a_1, q_0, S_1)(a_2, q_1, S_2) \dots (a_n, q_{n-1}, S_n)(\dashv, q_n, \emptyset)$.

The aim of the third transducer \mathcal{B} is to follow the output structure of T , which can be defined as follows: The *output structure* of a run is a labeled and directed graph such that, for each variable X useful at a position j , we have two nodes X_i^j and X_o^j linked by a path whose concatenated labels form the value stored in X at position j of the run. Formally, the output structure of a run $q_0 \xrightarrow{u} q_n$ is the oriented graph over $\mathcal{X} \times [1, |u|] \times \{i, o\}$ whose edges are labeled by output and are of the form:

- $((X, j, i), v, (Y, j-1, i))$ if $\rho(q_{j-1}, a_j, X)$ starts with vY .
- $((X, j, o), v, (Y, j-1, i))$ if there exists Z such that XvY appears in $\rho(q_{j-1}, a_j, Z)$,
- $((X, j, o), v, (Y, j+1, o))$ if $\rho(q_{j-1}, a_j, Y)$ ends by Xv ,
- $((X, j, i), v, (X, j+1, o))$ if $\rho(q_{j-1}, a_j, X) = v$.

We furthermore restrict to the connected component corresponding to the actual output of the run.

Now let $\mathcal{B} = (A \times Q \times 2^{\mathcal{X}}, B, P, p_0, \mu, \nu, \{f\})$ be defined by:

- $P = \mathcal{X} \times \{i, o\} \uplus \{p_0, f\}$ is the set of states. The transducer does a first left-to-right reading of the input in state p_0 . The subset $\mathcal{X} \times \{i, o\}$ will then be used to follow the output structure while keeping track of which variable we are currently producing. The set $\{i, o\}$ stands for *in* and *out* and corresponds to the similar notions in the output structure. Informally, *in* states will move to the left, while *out* states move to the right. The states p_0 and f are new states that are respectively initial and final.

The transition function $\mu : P \times A \times Q \times 2^{\mathcal{X}} \rightarrow P \times \{-1, 0, +1\}$ and the production function $\nu : P \times A \times Q \times 2^{\mathcal{X}} \rightarrow B^*$ are detailed below. In the following, we consider that the transducer is in state p reading the triplet $t = (a, q, S)$ or one of the endmarkers (see Figure 4).

- If $p = p_0$ and $a \neq \dashv$, then we set $\mu(p_0, t) = (p_0, +1)$ and $\nu(p_0, t) = \epsilon$.
- If $p = p_0$ and $a = \dashv$, then if $F(q)$ starts by uX with $u \in B^*$ and $X \in \mathcal{X}$, then $\mu(p, t) = ((X, i), -1)$ and $\nu(p, t) = u$.
- If $p = (X, i)$, and $t \neq \dashv$ then:
 - either $\rho(q, a)(X) = u \in B^*$ and does not contain any variable, and we set $\mu(p, t) = ((X, o), +1)$ and $\nu(p, t) = u$,
 - or $\rho(q, a)(X)$ starts by uY with $u \in B^*$ and $Y \in \mathcal{X}$, then $\mu(p, t) = ((Y, i), -1)$ and $\nu(p, t) = u$.
- If $p = (X, i)$, and $t = \dashv$ then $\mu(p, t) = ((X, o), +1)$ and $\nu(p, t) = \epsilon$.
- If $p = (X, o)$ and $a \neq \dashv$, then let Y be the unique variable of S such that X appears in $\rho(q, a)(Y)$. Then we have:
 - either $\rho(q, a)(Y)$ ends by Xu with u in B^* and we set $\mu(p, t) = ((Y, o), +1)$ and $\nu(p, t) = u$,
 - or $\rho(q, a)(Y)$ is of the form $(B \cup \mathcal{X})^* X u X' (B \cup \mathcal{X})^*$ and we set $\mu(p, t) = ((X', i), -1)$ and $\nu(p, t) = u$.

Note that the unicity of such Y in S is due to the 1-boundedness property. If T is copyless, then this information is irrelevant and \mathcal{A}'' can be bypassed.

- If $p = (X, o)$, $q \in Q_f$ and $a = \dashv$ then:
 - either $F(q)$ ends by Xu with u in B^* and we set $\mu(p, t) = (f, +1)$ and $\nu(p, t) = u$,
 - or $F(q)$ is of the form $(B \cup \mathcal{X})^* XuX'(B \cup \mathcal{X})^*$ and we set $\mu(p, t) = ((X', i), -1)$ and $\nu(p, t) = u$.

Then we can conclude the proof as $T = \mathcal{B} \circ \mathcal{A}'' \circ \mathcal{A}'$ and 2-way transducers are closed by composition [5].

Regarding complexity, a careful analysis of the composition of a one-way transducer of size m with a two-way transducer of size n from [6,4] shows that this can be done by a two-way transducer of size $O(nm^m)$. Then given a 1-bounded SST with n states and m variables, we can construct a deterministic two-way transducer of size $O(m(2^m)^{2^m} n^n) = O(m2^{m2^m} n^n)$. If T is copyless, the sequential right-to-left transducer can be omitted, and the resulting 2DFT is of size $O(mn^n)$.

Theorem 5. *Let T be an aperiodic 1-bounded SST. Then the equivalent 2DFT constructed using Theorem 4 is also aperiodic.*

Proof. We prove separately the aperiodicity of the three transducers. Then the result comes from the fact that aperiodicity is preserved by composition of a one-way by a two-way [4].

First, consider the transducer \mathcal{A}' . It is a one-way transducer that simply enriches the input word with transitions from T , each enrichment corresponding to the transition taken by T in the previous step. Then since T is aperiodic, so is its underlying automaton. Then the enrichment and thus \mathcal{A}' are aperiodic.

Secondly, given an input word, the transducer \mathcal{A}'' stores at each position the set of variables that will be output by T . Now as T is aperiodic, the flow of variable is aperiodic. Thus the value taken by this set is aperiodic and so is \mathcal{A}'' .

Now, consider the transducer \mathcal{B} and a run r of \mathcal{B} over u^n starting in state p . Note that the fact that there exists a run over an enriched input word v implies that it is well founded, meaning that it is the image of some word of A^* by $\mathcal{A}'' \circ \mathcal{A}'$. If p is of the form (X, i) , then the run starts from the right of u^n and follows the substitution $\sigma_r(X)$. It exits u^n either in state (X, o) on the right if $\sigma_r(X)$ is a word of B^* , or in a state (Y, i) on the left where Y is the first variable appearing in $\sigma_r(X)$. In both cases the state at the end of the run only depends on the underlying automata of T and the order of variables appearing in the substitution induced by the run. Since the substitution transition monoid is aperiodic if T is aperiodic by Theorem 3, and then a similar run exists over u^{n+1} .

Finally, if p is of the form (X, o) , then the state in which the run exits u^n depends on the unique variable Y such that X belongs to $\sigma_r(Y)$ and Y belongs to the set of variables of the last letter of the input. Then the run follows the substitution $\sigma_r(Y)$. It will exit the input word in state (X', i) on the left if XX' appears in $\tilde{\sigma}_r(Y)$ for some variable X' and in state (Y, o) otherwise. As the flow

of variable as well as the underlying automaton are aperiodic, a similar run exists over u^{n+1} .

We conclude the proof by noticing that the same arguments will hold to reduce runs over u^{n+1} to runs over u^n .

From 2DFT to copyless SST

Theorem 6. *Let T be a 2DFT with n states. Then we can effectively construct a copyless SST with $O((2n)^{2n})$ states and $2n - 1$ variables that computes the same function.*

Let $T = (A, B, Q, i, \delta, \gamma, F)$ be a 2DFT. Let us suppose that the transducer T starts to read its input from the end, and not from the beginning, i.e., given an input w , the initial configuration is $(q_0, |w| + 1)$. Moreover, let us suppose that for any transition (p, \vdash, q, m) of T , $\gamma(p, \vdash, q, m) = \epsilon$. Note that any 2DFT can be transformed with ease into a transducer satisfying those two properties.

In order to reproduce the behavior of T with an SST T' , we need to keep track of the right-to-right runs of T . Moreover, as we want T' to be copyless, it is not possible to store the production of a right-to-right run into a single variable, since two different runs might share a common suffix, and require copy in order to update the corresponding variables. This leads us to modelize the right-to-right runs of T with rooted forests whose vertices are included into $2^Q \setminus \emptyset$. The idea is that, given two states $q_1, q_2 \in Q$, a merging between the right-to-right runs starting from q_1 and q_2 can be represented by adding an edge from both $\{q_1\}$ and $\{q_2\}$ towards $\{q_1, q_2\}$. Formally, we use the set \mathcal{F}_Q of rooted forests $G = (V, E)$ such that V is a subset of $2^Q \setminus \emptyset$, and the following properties are satisfied.

- The roots of G are disjoint subsets of Q .
- For every vertex s , the sons of s are disjoint proper subsets of s .

Note that two graphs of \mathcal{F}_Q with same set of vertices are equal, hence each element of \mathcal{F}_Q is uniquely defined by its set of vertices. In order to also keep track of the target states of the right-to-right runs, the states of T' are pairs (G, ϕ) , where $G \in \mathcal{F}_Q$, and ϕ is an injective function mapping each tree of G , corresponding to a set of merging runs, to an element of Q , corresponding to the target state of those runs. In the following definitions, for every vertex v of G we will usually denote by $\phi(v)$ the state $\phi(T_v)$, where T_v is the tree containing the vertex v .

For every word w over the alphabet $\bar{A} = A \cup \{\vdash, \dashv\}$, we now expose an inductive construction of a pair (G_w, ϕ_w) , where $G_w \in \mathcal{F}_Q$ and ϕ_w maps the roots of G_w to Q , that contains all the information concerning the right-to-right runs of T over the input w , and their mergings. Formally, for every state $q \in Q$ that appears at least in one vertex of G_w , let $s_q \in 2^Q$ be the vertex of G_w of minimal size such that $q \in s_q$. Then we want the following property to be satisfied.

P₁ For all $(p, q) \in \text{bh}_{rr}(w)$, we have $\phi_w(s_p) = q$.

Let $G_\epsilon \in \mathcal{F}_Q$ be the graph on 0 vertex, and let $\phi_\epsilon : V \rightarrow Q$ be the empty function. Note that **P₁** is trivially satisfied for $w = \epsilon$, as $\text{bh}_{rr}(\epsilon)$ is empty, by definition. Now, let $w \in \bar{A}^*$, let $a \in \bar{A}$, and suppose that (G_w, ϕ_w) is defined such that **P₁** is satisfied for w . Then (G_{wa}, ϕ_{wa}) is built based on (G_w, ϕ_w) in three steps. First, we build a graph G'_{wa} by adding to G_w edges corresponding to the function ϕ_w . Second, we build a graph G''_{wa} by adding to G'_{wa} the local transitions induced by the letter a . Finally, we reduce G''_{wa} into an element $G_{wa} = (V_{wa}, E_{wa})$ of \mathcal{F}_Q .

- Let $G'_{wa} = (V'_{wa}, E'_{wa})$ be the graph defined by $V'_{wa} = V_w \cup Q^\ddagger$, where Q^\ddagger is a copy of the set Q , and $E'_{wa} = E_w \cup \{(\phi_w^{-1}(p), p^\ddagger) \in 2^Q \times Q^\ddagger \mid p \in \text{Im}(\phi_w)\}$. Since **P₁** is satisfied for w by supposition, for every $(p, q) \in \text{bh}_{rr}(w)$ there is a path in G'_{wa} between s_p and q^\ddagger .
- Let $G''_{wa} = (V''_{wa}, E''_{wa})$ be the graph defined by $V''_{wa} = V'_{wa} \cup Q^\circ$, where Q° is a copy of the set Q , and let $E''_{wa} = E'_{wa} \cup \{(p^\ddagger, \tau(p)) \in Q^\ddagger \times V'_{wa} \mid p \in Q\}$, where

$$\tau(p) = \begin{cases} q^\circ \in Q^\circ & \text{if } \delta(p, a) = (q, +1), \\ q^\ddagger \in Q^\ddagger & \text{if } \delta(p, a) = (q, 0), \\ s_q \in V_w & \text{if } \delta(p, a) = (q, -1) \text{ and } s_q \text{ is the smallest vertex containing } q. \end{cases}$$

For every $(p, q) \in \text{bh}_{rr}(wa)$, there exists a path in G'_{wa} between p^\ddagger and q° .

- Let $\text{in}(s) : V''_{wa} \rightarrow 2^Q$ be the function mapping each vertex s of G''_{wa} to the set of states q such that there exists a path from q^\ddagger to s in G''_{wa} . Moreover, let $\text{out}(s) : V''_{wa} \rightarrow Q \cup \{\perp\}$ be the function mapping each vertex s of G''_{wa} , to \perp if the unique path starting from s loops infinitely, and to $q \in Q$ if the target of this path is the vertex $q^\circ \in Q^\circ$. Let $G_{wa} = (V_{wa}, E_{wa})$, where $V_{wa} = \{\text{in}(s) \subset Q \mid s \in V''_{wa}, \text{out}(s) \neq \perp\}$, and let ϕ_{wa} be the function mapping each vertex $\text{in}(s)$ of V_{wa} to $\text{out}(s)$. For every $(p, q) \in \text{bh}_{rr}(wa)$, since $\text{out}(p^\ddagger) = q$, $\text{in}(p^\ddagger)$ is a vertex of G_{wa} , and $\phi_{wa}(\text{in}(p^\ddagger)) = \text{out}(p^\ddagger) = q$, hence **P₁** is satisfied, as $\text{in}(p^\ddagger) = s_p$.

Remark 1. Since the construction of G_{wa} only depends on G_w and a , for every $w' \in \bar{A}^*$ such that $G_{w'} = G_w$, we have $G_{w'a} = G_{wa}$.

Remark 2. By construction, for each subset s of Q , s is a vertex of the graph G_w if and only the subset of right-to-right runs of T over w containing the runs whose starting state belongs to s merge at some point, before merging with any other.

The next results will allow us to obtain the bound over the size of the set of states presented in the statement of Theorem 6.

Lemma 2. *Every rooted forest $G = (V, E)$ in \mathcal{F}_Q has at most $2|Q| - 1$ vertices.*

Proof. This is proved by induction over $|Q|$. If $|Q| = 1$, $|2^Q \setminus \emptyset| = 1$, hence $|V| \leq 1 = 2|Q| - 1$. Now suppose that $|Q| > 1$, and that the the result is true for every set Q' such that $|Q'| < |Q|$. Let G' be the graph obtained by removing all the roots of G . Then G' is a union of trees $G_1 = (V_1, E_1), G_2 = (V_2, E_2), \dots, G_m = (V_m, E_m)$. For every $1 \leq i \leq m$, the forest G_i is an element of \mathcal{F}_{s_i} , where $s_i \subset Q$ denotes the root of G_i . Therefore, by using the induction hypothesis, we have

$$\begin{aligned} |V| &= 1 + |V_1| + |V_2| + \dots + |V_m| \\ &\leq 1 + (2|s_1| - 1) + (2|s_2| - 1) + \dots + (2|s_m| - 1) \\ &= 2(|s_1| + \dots + |s_m|) + 1 - m \\ &\leq 2|Q| - 1. \end{aligned}$$

Lemma 3. *Let $n = |Q|$. The size of \mathcal{F}_Q is smaller than or equal to $\frac{(2n)^{2n-2}}{(n-2)!}$.*

Proof. By Cayley's Formula, there exists exactly $(2n)^{2n-2}$ labeled trees on $2n$ vertices. The result follows from the fact that any element of \mathcal{F}_Q can be represented by a tree on $2n$ vertices, of which at most $n+2$ are labeled, which justifies the denominator, as the labels of $(n-2)$ vertices can be forgotten.

Given an element $G = (V, E)$ of \mathcal{F}_Q , we know that $|V| \leq 2n - 1$ by Lemma 2. Let G' be the tree on $2n$ vertices obtained by adding to G a vertex s_{\perp} , an edge from each root of G to s_{\perp} , and, if $|V| < 2n - 1$, a linear path composed of $2n - |V| - 1$ new vertices starting from a vertex s_{\top} , and whose end is linked to s_{\perp} . Then G can be computed back from any graph isomorphic to G' , as long as we set the label \perp and \top to the vertices corresponding to s_{\perp} and s_{\top} , and the label q to the vertex corresponding to s_q , where s_q is the smallest vertex of V containing q , for every q appearing in a vertex of G .

Corollary 3. *Let $n = |Q|$. The size of $\{(G_w, \phi_w) | w \in \bar{A}^*\}$ is smaller than or equal to $(2n)^{2n}$.*

Proof. For every $w \in \bar{A}^*$, since ϕ_w is an injective function mapping the trees of G_w to Q , and G_w contains at most n trees,

$$|\{(G_w, \phi_w) | w \in \bar{A}^*\}| \leq n! |\mathcal{F}_Q| \leq (2n)^{2n}$$

An other consequence of Lemma 2 is that for every word $w \in \bar{A}^*$, the graph $G_w = (V_w, E_w)$ admits an injective vertex labeling $\lambda_w : V_w \rightarrow \mathcal{X}$, where $\mathcal{X} = (X_1, \dots, X_{2|Q|-1})$ is a set containing $2|Q|-1$ variables. We shall now present for every $w \in \bar{A}^*$, the construction of a substitution $\sigma_w \in \mathcal{S}_{\mathcal{X}, B}$ that will allow us, together with the graph G_w and its vertex labeling λ_w , to describe the output production of the right-to-right runs of T over w . Formally, for every $(p, q) \in \text{bh}_{rr}(w)$, let $w_{p,q} \in B^*$ denote the production of the corresponding right-to-right run. Moreover, for every vertex s of G_w , let $\bar{\lambda}_w(s)$ denote the concatenation of the λ -labels of the vertices forming the path starting from the s in G_w , and for every state $q \in Q$ that appears at least in one vertex of G_w , let $s_q \in 2^Q$ be the vertex of G_w of minimal size such that $q \in s_q$. Then we want the following property to be satisfied.

P₂ For all $(p, q) \in \text{bh}_{rr}(w)$, we have $(\sigma_w)(\bar{\lambda}_w(s_p)) = w_{p,q}$.

Let σ_ϵ be the substitution mapping each variable to ϵ . Once again, since $\text{bh}_{rr}(w)$ is empty, **P₂** is trivially satisfied for $w = \epsilon$. Let $w \in \bar{A}^*$, let $a \in \bar{A}$, and let us suppose that **P₂** is satisfied for w . The substitution σ_{wa} is defined as the composition of a substitution $\sigma_{w,a}$, whose construction we will now present, with σ_w . In order to build $\sigma_{w,a}$, we define a vertex labeling $\mu_{wa} : V_{wa} \rightarrow (\mathcal{X} \cup B)^*$. We require μ_{wa} to be copyless, i.e., for any variable $X \in \mathcal{X}$, there exists at most one vertex s such that X occurs in $\mu_{wa}(s)$. Then, we define $\sigma_{w,a}$ as the copyless substitution mapping $\lambda_{wa}(s)$ to $\mu_{wa}(s)$. The labeling μ_{wa} is obtained by first extending the labeling λ_w of V_w to a labeling μ''_{wa} of V''_{wa} , and then reducing it to G_{wa} . We denote by $\bar{\mu}_{wa}(s)$ (resp. $\bar{\mu}''_{wa}(s)$) the concatenation of the labels of the vertices forming the path starting from a vertex s of G_{wa} (resp. G''_{wa}).

- Let $\mu''_{wa} : V''_{wa} \rightarrow \mathcal{X} \cup B^*$ be the substitution mapping $s \in V_w$ to $\lambda_w(s)$, $q^\circ \in Q^\circ$ to ϵ , and $q^\natural \in Q^\natural$ to $\gamma(p, a, q, m) \in B^*$, where $\delta(p, a) = (q, m)$. Since λ_w is injective, this labeling is copyless. Moreover, since **P₂** is satisfied for w by supposition, by definition of G''_{wa} we have, for every $(p, q) \in \text{bh}_{rr}(wa)$,

$$\sigma_w(\bar{\mu}''_{wa}(p^\natural)) = w_{p,q}.$$

- For every $t \in V_{wa}$, the set of vertices s of G''_{wa} such that $\text{in}(s) = t$ is not empty, and they form a path s_1, \dots, s_m . Let $\mu(s) = \mu'(s_1) \dots \mu'(s_m)$. Since μ''_{wa} is copyless, so is μ_{wa} . Moreover,

$$(\sigma_w)(\bar{\mu}_{wa}(\text{in}(p^\natural))) = w_{p,q}.$$

Since $\sigma_{wa} = \sigma_w \circ \sigma_{w,a}$, $\sigma_{w,a}(\lambda_{wa}(s)) = \mu_{wa}(s)$ by definition, and $\text{in}(p^\natural) = s_p$, this proves that **P₂** is satisfied for wa .

Remark 3. Since the construction of $\sigma_{w,a}$ only depends on G_w and a , for every $w' \in \bar{A}^*$ such that $G_{w'} = G_w$, we have $\sigma_{w',a} = \sigma_{w,a}$.

We are now ready to define formally the copyless SST $T' = (A, B, P, j, Q_f, \alpha, \mathcal{X}, \beta, F')$.

- $P = \{(G_{\vdash w}, \phi_{\vdash w}) | w \in A^*\}$,
- $j = (G_{\vdash}, \phi_{\vdash})$,
- $Q_f = \{(G_{\vdash w}, \phi_{\vdash w}) | \phi_{\vdash w \dashv}(i) \in F\}$,
- $\alpha : P \times A \rightarrow P, ((G_{\vdash w}, \phi_{\vdash w}), a) \mapsto (G_{\vdash wa}, \phi_{\vdash wa})$, which is well-defined, by Remark 1,
- $\mathcal{X} = \{X_i | 1 \leq i \leq 2|Q| - 1\}$,
- $\beta : P \times A \rightarrow \mathcal{S}_{\mathcal{X}, B}, ((G_{\vdash w}, \phi_{\vdash w}), a) \mapsto \sigma_{\vdash w, a}$, which is well-defined, by Remark 3,
- $F' : Q_f \rightarrow (\mathcal{X} \cup B)^*, (G_{\vdash w}, \phi_{\vdash w}) \mapsto \sigma_{\vdash w \dashv}(\phi_{\vdash w \dashv}(i))$.

The state reached by T' over an input word $w \in A^*$ is $G_{\vdash w}$, and the substitution induced by the corresponding run is $\sigma_{\vdash w}$. Therefore, by **P₁** and the definition of Q_f , the domain of the functions defined by T and T' are identical. Moreover, since σ_{\vdash} is the substitution mapping all the variables to ϵ , by supposition, we have, by **P₂** and the definition of F' , that the image of a given word by those two functions are also identical, hence the functions are the same.

Theorem 7. *Let T be an aperiodic 2DFT. Then the equivalent SST constructed using Theorem 6 is also aperiodic.*

To prove this theorem, we introduce a new equivalence relation, and prove that the aperiodicity of the 2DFT implies aperiodicity of this relation. The aperiodicity of the SST then follows from this. We say that two words v and w are *merge equivalent* ($v \sim_m w$) if they induce the same merges in the same order in their four behavior relations. Let us remark that if $v \sim_m w$, then for any word u , $G_{uv} = G_{uw}$. This is due to the fact that G_{uv} represents the merges of the right-to-right runs over uv , and these runs can be decomposed in right-to-right runs over u and partial runs over v .

Lemma 4. *Let $w \in \bar{A}^*$ be such that $w^n \sim_T w^{n+1}$. Then $w^n \sim_m w^{n+1}$.*

Proof. Let w and n be such that $w^n \sim_T w^{n+1}$. By definition, they model the same partial runs, and thus the same merges appear. Thus we only need to prove that they appear in the same order. Consider two merges that appear consecutively in w^n . We prove that they appear in the same order in w^{n+1} , depending on the kind of partial run that we consider. If the merges are of right-to-right or left-to-left runs, then the exact same runs appear in w^{n+1} and thus the same merges appear. If they affect right-to-left (resp. left-to-right) runs, then the rightmost (resp. leftmost) n iterations of w in w^{n+1} will merge these runs. By noticing that after merging, two runs can not be separated again, the two merges will appear in the same order in w^{n+1} , concluding the proof.

The two following lemmas now conclude the proof, since the aperiodicity of both the underlying automaton and the substitution imply the aperiodicity of the whole SST.

Lemma 5. *Let $w \in \bar{A}^*$ be such that $w^n \sim_T w^{n+1}$. Then for every $u \in \bar{A}^*$, $G_{uw^{n+1}} = G_{uw^n}$.*

Proof. This lemma comes directly from Lemma 4 and the previous remark stating that the merge equivalence implies the equivalence for the underlying automaton of the SST.

Lemma 6. *Let $w \in \bar{A}^*$ be such that $w^n \sim_T w^{n+1}$. Then for every $u \in \bar{A}^*$, $\sigma_{u,w^n} \sim \sigma_{u,w^{n+1}}$.*

Proof. By construction, reducing a graph from the proof of Theorem 6 G'_{ua} to G_{ua} amounts to delete the unnecessary information from G_u , i.e. deleting cycles and reducing paths with no new merges to a single vertex. Thus each vertex from G_u can either be traced back to a vertex of G_{ua} or is deleted. Should we forget about the production, the flow of variables then corresponds exactly to this. Thanks to Lemma 4, we know that w^n and w^{n+1} are merge equivalent. Then given a state G_u and one of its vertex, it can be traced back to the same vertex after reading w^n or w^{n+1} . Since we have a unique way of mapping vertices to variables, the substitutions σ_{uw^n} and $\sigma_{uw^{n+1}}$ will be equal when production is erased, proving the aperiodicity of the substitution function.

From k -bounded to 1-bounded SST

Theorem 8. *Given a k -bounded SST T with n states and m variables, we can effectively construct an equivalent 1-bounded SST. This new SST has $n2^N$ states and mkN variables, where $N = O(n^n(k+1)^{nm^2})$ is the size of the flow transition monoid M_T .*

Proof. In order to move from a k -bounded SST to a 1-bounded SST, the natural idea is to use copies of each variable. However, we cannot maintain k copies of each variable all the time: suppose that X flows into Y and Z , which both occur in the final output. If we have k copies of X , we cannot produce in a 1-bounded way k copies of Y and k copies of Z . We will thus limit, for each variable X , the number of copies of X we maintain. In order to get this information, we will use a look-ahead information on the suffix of the run.

The proof relies on the following fact: suppose that we know at each step what is the substitution induced by the suffix of the run. From this substitution, for each variable X we know the value of the integer n such that X will be involved exactly n times in the final output. We can thus copy each variable sufficiently many times and use them to produce this substitution in a copyless fashion.

One can observe that there are finitely many substitutions, this information being held in the transition monoid of the SST. Then we can compute, at each step and for each possible substitution, a copyless update. But as a given element of the monoid may have several successors, the update function flows variables from one element to variables of several elements. As these variables are never recombined, we get the 1-boundedness of the construction.

Let $T = (A, B, Q, q_0, Q_f, \delta, \mathcal{X}, \rho, F)$ be an aperiodic k -bounded SST, M_T be its transition monoid, and $\eta_T : A^* \rightarrow M_T$ be its transition morphism.

We construct $T' = (A, B, Q', q'_0, Q'_f, \delta', \mathcal{X}', \rho', F')$ where:

- The set of states $Q' = Q \times \mathcal{P}(M_T)$ is the current state plus a set of elements of M_T corresponding to the possible images of the current suffix.
- $q'_0 = (q_0, S_0)$ where $S_0 = \{m \in M_T \mid \delta(q_0, m) \in Q_f\}$ is the set of relevant possible images of input words. Here, we are abusing notations as $\delta(q, m)$ stands for $\delta(q, u)$ where $\eta_T(u) = m$. By definition of the transition monoid, we have that $\eta_T(u) = \eta_T(v)$ implies $\delta(q, u) = \delta(q, v)$, thus this is well defined.
- $Q'_f = \{(q, S) \mid q \in F \text{ and } 1_{M_T} \in S\}$.
- $\delta' : Q' \times A \rightarrow Q'$ is defined by $\delta'((q, S), a) = (q', S')$ where $q' = \delta(q, a)$ and $S' = \{m \in M_T \mid \eta_T(a)m \in S\}$.
- $\mathcal{X}' = \mathcal{X} \times M_T \times \{1, \dots, k\}$. Variables from \mathcal{X}' will be denoted X_i^m for $X \in \mathcal{X}, i \leq k$ and $m \in M_T$.
- The variable update function is defined as follows. First given a state q of T and an element m of M_T , we define $\tilde{\sigma}_{q,m}$ as the projection of the output substitution induced by a run starting on q on a word whose image is m , i.e. $\tilde{\sigma}_{q,u} = \gamma(q, u) \circ F(\delta(q, u))$ for $\eta_T(u) = m$. Note that by definition of the transition monoid of an SST, it is well defined.

Now consider a transition $(q, S) \xrightarrow{a} (q', S')$, $n \in S'$ and $0 < i \leq |\tilde{\sigma}_{q',n}|_X$, $\rho'((q, S), a, X_i^n)$ is defined similarly to $\rho(q, a, X)$, where all variables are labeled by the element $\eta_T(a)n$ and numbered to ensure the 1-bounded property. Such a numbering is possible thanks to the fact that n indicates which variables are used as well as how many times. This allows us to copy each variables the right amount of times, using different copies at each occurrence. The k -bounded property then ensures that we will never need more than k variables for a possible output.

- $F' : Q'_f \rightarrow (B \cup \mathcal{X}')^*$ is defined as follows. Let $(q, S) \in Q'_f$. The string $F'(q, S)$ is obtained from the string $F(q)$ by substituting each variable X by a variable X_i^n , where $0 < i \leq |F(q)|_X$ and $n = 1_{M_T}$.

Theorem 9. *Let T be an aperiodic k -bounded SST. Then the equivalent 1-bounded SST constructed using Theorem 8 is also aperiodic.*

Proof. We now have to prove that T' is aperiodic. We claim that the runs of T' are of the form $(q, S) \xrightarrow{u} (q', S')$ where $q \xrightarrow{u} T q'$ and $S' = \{n \in M_T \mid \eta_T(u)n \in S\}$, which holds by construction of T' . The update for such a run is then the update of T over the run $q \xrightarrow{u} T q'$, where variables are labeled by elements from S and S' and numbered accordingly. Then as T is aperiodic, the Q part of the run is also aperiodic by construction. The other part computes sets of runs according to M_T , which is also aperiodic. Then T' will also be aperiodic as the set S' only depends on the image of the word read, and by definition $\eta_T(u^n) = \eta_T(u^{n+1})$ for n large enough.